Consultations statistiques avec le logiciel R

Peut-on modifier la fonction `coinertia` pour utiliser des très grands nombres de variables ?

Résumé

La question est posée par des utilisateurs de l'analyse de co-inertie en biologie moléculaire. La fonction coinertia est modifiée pour prendre en compte des couples de tableaux où le nombre de colonnes est beaucoup plus grand que le nombre de lignes de chaque côté.

Plan

1. Origine de la question

La question était apparue dans l'article de Thioulouse et Lobry (1995) :

> *Computations and graphical displays were obtained using the ADE package (Chessel and Dolédec,(1993); Thioulouse* et al. *(1994)). All computations were performed with ADE version 4.0 on an Apple PowerMacintosh 8100/80 with 16 megabytes RAM (random access memory). The data type for floating point variables was long double (10 bytes). Computation times were 10 seconds for the PCA, 15 seconds for the CA, and 56 minutes for the co-inertia analysis using MC680x0 microprocessor emulation. Using a native compiler (generating PowerPC601 microprocessor code) reduced the computation time 10-fold (about 1 second for PCA and CA, 5 minutes for co-inertia analysis). Improvements in the algorithm should provide computation times for co-inertia analysis comparable to those for PCA and CA: the matrix from which eigenvalues and eigenvectors are computed will be of dimension min(n, p, q), with n = number of observations, p = number of variables in the first table, q = number of variables in the second table* ***(instead of min(p, q) as is now the case)****.*

Dans cet article, en effet, pour la première fois un nombre considérable de variables est utilisé dans chacun des deux tableaux d'une co-inertie. A partir du site de Jean Lobry :

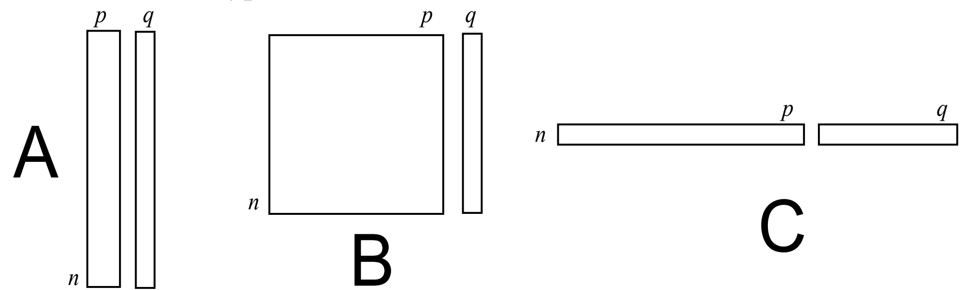
http://pbil.univ-lyon1.fr/members/lobry/

les données sont accessibles à

ftp://pbil.univ-lyon1.fr/pub/datasets/CABIOS95/

File Name	Number of rows	Number of columns	Content
AANames	20	1	Amino-acid 3-letter code
ProtNames	999	1	Proteins names
IndiceNames	402	1	Indices names
IndiceVals	402	20	Indices values
ProtComp	999	20	Proteins composition

Le premier tableau, **ProtNames**, compte 20 lignes (acides aminés) et 999 (protéines codées par les gènes de *E coli*) et le second, **IndiceVals**, compte 20 lignes (acides aminés) et 402 variables, compilation de 402 propriétés biochimiques et physicochimiques de ces acides aminés. Ils sont présentés transposés pour des raisons pratiques. En général les tableaux sont appariés par les lignes et on a donc une situation du type C :



La modification prévue n'avait pas encore été faite, quand la question s'est à nouveau posée dans Culhane et al. (2003) qui utilise des tableaux à 60 lignes et 200 à 5000 colonnes par tableaux. Lors des tests de permutations, des centaines d'analyses sont faites et la question de l'optimisation de la procédure est de grand intérêt. C'est un exercice intéressant qui utilise le schéma de dualité.

Pour tester le programme, nous faisons une expérience en rajoutant 50 variables environnementales aléatoires et 100 espèces aléatoires à un couple de tableau bien connu :

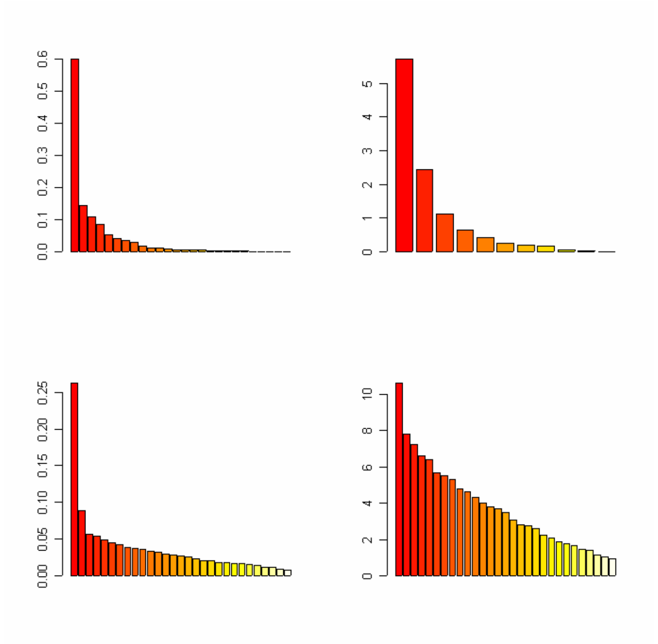
```
data(doubs)
X=doubs$mil
dim(X)
[1] 30 11
X=cbind.data.frame(X,mvrnorm(30,rep(0,100),diag(1,100)))
dim(X)
[1]  30 111
names(X)
  [1] "das" "alt" "pen" "deb" "pH"  "dur" "pho" "nit" "amm" "oxy" "dbo" "1"
 [13] "2"   "3"   "4"   "5"   "6"   "7"   "8"   "9"   "10"  "11"  "12"  "13"
...
 [97] "86"  "87"  "88"  "89"  "90"  "91"  "92"  "93"  "94"  "95"  "96"  "97"
[109] "98"  "99"  "100"

w=lapply(1:150,function(x) {a=runif(1);return(rbinom(30,1,a))})
w=as.data.frame(w)
w=w[,apply(w,2,sum)>0]
dim(w)
[1]  30 141
names(w)=paste("w",1:141,sep="")
Y = cbind.data.frame(Y,w)

dim(X)
[1]  30 111 11 variables expérimentales + 100 variables aléatoires
dim(Y)
[1]  30 168 27 espèces de poisson + 141 espèces aléatoires

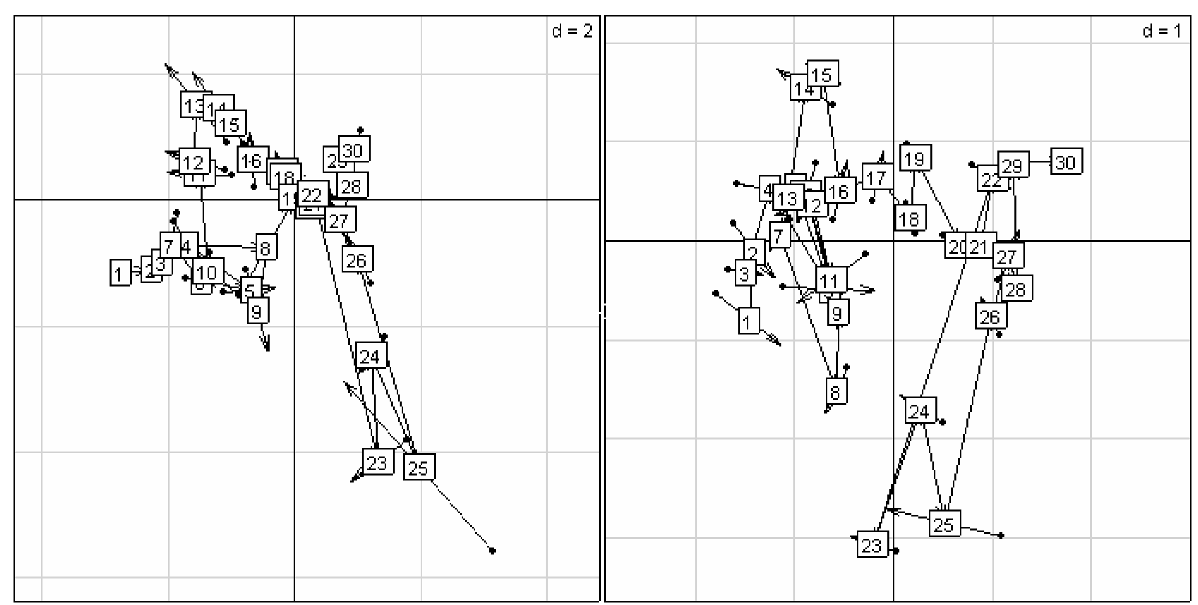
par(mfrow=c(2,2))
coa1=dudi.coa(doubs$poi)
pca1=dudi.pca(doubs$mil,coa1$lw)
coa2=dudi.coa(Y)
pca2=dudi.pca(X,coa2$lw)
```

On garde partout deux axes.



```
coi1=coinertia(pca1,coa1)
Select the number of axes: 2
coi2=coinertia(pca2,coa2)
Select the number of axes: 2
s.match(coi1$mX,coi1$mY)
s.traject((coi1$mX+coi1$mY )/2,add.p=T,clab=0)
s.match(coi1$mX,coi1$mY,add.p=T)

s.match(coi2$mX,coi2$mY)
s.traject((coi2$mX+coi2$mY )/2,add.p=T,clab=0)
s.match(coi2$mX,coi2$mY,add.p=T)
```



L'analyse est perturbée mais la structure principale est conservée. L'exemple permet de contrôler la modification de procédure.

2. Le schéma de la co-inertie

La version habituelle associe deux triplets $\left(\mathbf{X}, \mathbf{D}_p, \mathbf{D}_n\right)$ et $\left(\mathbf{Y}, \mathbf{D}_q, \mathbf{D}_n\right)$ où le premier tableau a p colonnes, le second a q colonnes et les deux ont en commun n lignes. Le schéma de la co-inertie est décomposé :

$$\begin{array}{ccc} \boxed{p} & \overset{\mathbf{D}_p}{\rightarrow} & \boxed{p} \\ \mathbf{Z}^t = \mathbf{X}^t\mathbf{D}_n\mathbf{Y} \uparrow & & \downarrow \mathbf{Z} = \mathbf{Y}^t\mathbf{D}_n\mathbf{X} \\ \boxed{q} & \underset{\mathbf{D}_q}{\leftarrow} & \boxed{q} \end{array} \Leftrightarrow \begin{array}{ccc} \boxed{p} & \overset{\mathbf{D}_p}{\rightarrow} & \boxed{p} \\ \mathbf{X}^t \uparrow & & \downarrow \mathbf{X} \\ \boxed{n} & & \boxed{n} \\ \mathbf{D}_n \uparrow & & \downarrow \mathbf{D}_n \\ \boxed{n} & & \boxed{n} \\ \mathbf{Y} \uparrow & & \downarrow \mathbf{Y}^t \\ \boxed{q} & \underset{\mathbf{D}_q}{\leftarrow} & \boxed{q} \end{array}$$

En général si **A**, **B** et **C** sont des matrices $p \times q$, $r \times p$ et $q \times r$, les matrices produits **CBA**, **BAC** et **ABC** ont un sens et sont carrées. Elles ont les mêmes valeurs propres non nulles car :

$$\mathbf{CBAu} = \lambda\mathbf{u} \Rightarrow \begin{Bmatrix} \mathbf{ACBv} = \lambda\mathbf{v} \\ \mathbf{v} = \mathbf{Au} \end{Bmatrix} \Rightarrow \begin{Bmatrix} \mathbf{BACw} = \lambda\mathbf{w} \\ \mathbf{w} = \mathbf{Bv} \end{Bmatrix} \Rightarrow \begin{Bmatrix} \mathbf{CBAu}_\circ = \lambda\mathbf{u}_\circ \\ \mathbf{u}_\circ = \mathbf{Cw} \end{Bmatrix}$$

On prendra donc la diagonalisation de la matrice non symétrique :

$$\mathbf{W} = \mathbf{D}_n\mathbf{X}\mathbf{D}_p\mathbf{X}^t\mathbf{D}_n\mathbf{Y}\mathbf{D}_q\mathbf{Y}^t \Rightarrow \mathbf{WU} = \mathbf{U\Lambda}$$

L'élimination des valeurs propres nulles se fait à un endroit quelconque (leur nombre varie, mais les valeurs propres non nulle sont invariantes). Les composantes principales de la co-inertie sont alors les vecteurs $\mathbf{Y}^t\mathbf{u}_k \big/ \left\|\mathbf{Y}^t\mathbf{u}_k\right\|_{\mathbf{D}_k}$. Ces vecteurs renormés à la racine de la valeur propre correspondante donneront les coordonnées des colonnes de **Y** et la suite de la procédure sera celle de la co-inertie

ordinaire. La nouvelle procédure est écrite dans une fonction indépendante puis intégrée, après vérification de 'identité des résultats, à la fonction courante. La modif est transparente à l'utilisateur.

3. Une expérience intéressante

```
coi1=coinertia(pca1,coa1)
Select the number of axes: 2
coi2=coinertia(pca2,coa2)
Select the number of axes: 2
```

```
summary(coi1)

Eigenvalues decomposition:
     eig  covar   sdX    sdY   corr
1 2.3416 1.5302 2.366 0.7591 0.8519
2 0.1750 0.4183 1.533 0.3336 0.8176

Inertia & coinertia X:
   inertia   max  ratio
1    5.598 5.728 0.9775
12   7.950 8.154 0.9750

Inertia & coinertia Y:
   inertia    max  ratio
1   0.5763 0.6010 0.9589
12  0.6876 0.7454 0.9225

RV:
 0.6363
```

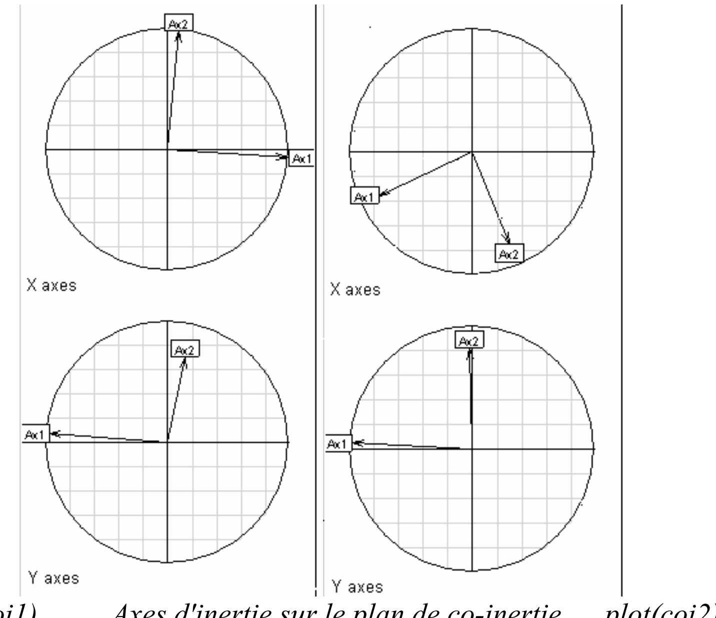
```
summary(coi2)

Eigenvalues decomposition:
     eig  covar   sdX    sdY   corr
1 1.7927 1.3389 2.934 0.5027 0.9077
2 0.5078 0.7126 2.716 0.2777 0.9448

Inertia & coinertia X:
   inertia   max  ratio
1     8.61 10.61 0.8111
12   15.99 18.43 0.8675

Inertia & coinertia Y:
   inertia    max  ratio
1   0.2527 0.2620 0.9646
12  0.3298 0.3514 0.9385

RV:
 0.6275
```



plot(coi1) *Axes d'inertie sur le plan de co-inertie* *plot(coi2)*

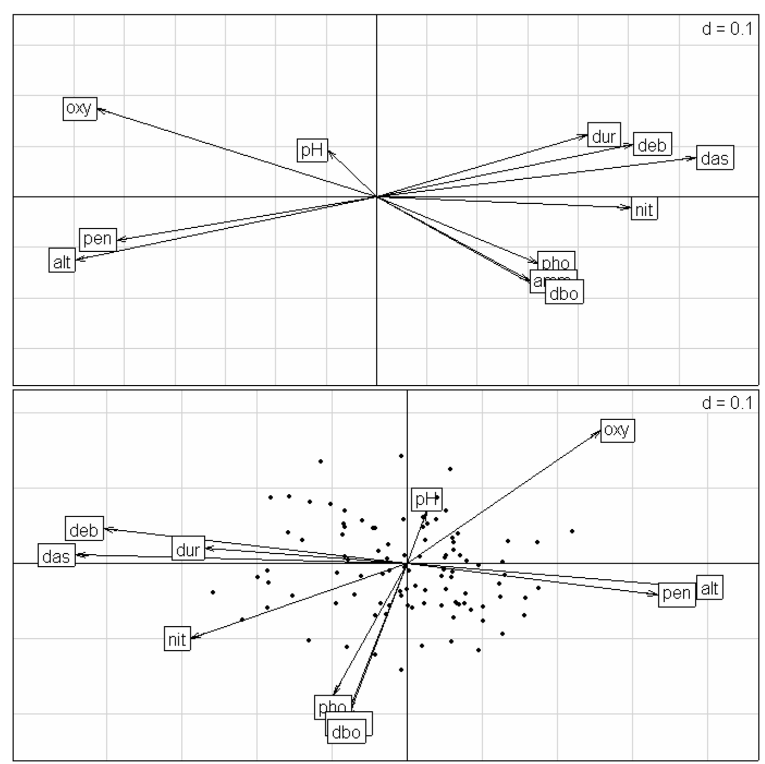
```
cor(coi1$lX,coi2$lX)
         AxcX1   AxcX2
AxcX1 -0.86233 -0.5303
AxcX2 -0.01833  0.8697
 cor(coi1$lY,coi2$lY)
         AxcY1   AxcY2
AxcY1 -0.86000 -0.2559
AxcY2 -0.06289  0.7942
```

En dépit de l'introduction massive de variabilité aléatoire dans les données (50 variables et 100 espèces sur 11 variables et 30 espèces mesurées !) l'analyse retrouve les scores initiaux, certes avec du brouillage, mais les extrait quand même d'un mélange aussi pourri.

Les points sont les espèces fictives dans l'analyse simulée

```
par(mfrow=c(2,1))
s.label(coi1$li)
```

```
s.label(coi2$li[1:27,])
s.label(coi2$li[-(1:27),],clab=0,add.p=T)
```

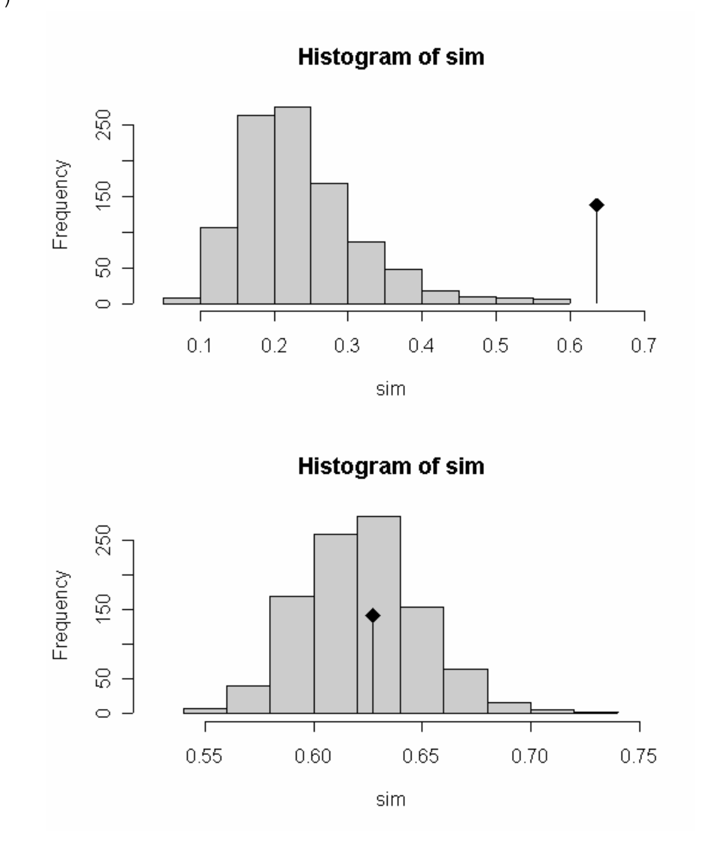


Les points sont les variables fictives dans l'analyse simulée

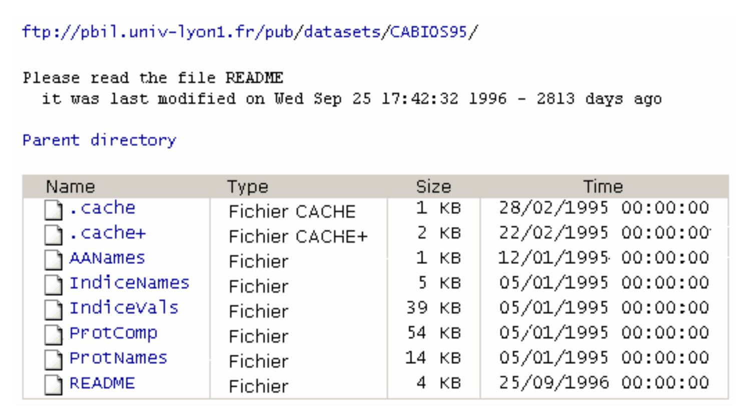
```
s.arrow(coi1$co)
s.arrow(coi2$co[1:11,])
s.label(coi2$co[-(1:11),],clab=0,add.p=T)
```

Mais si le premier test de permutations est très significatif, le second ne l'est pas :

```
plot(randtest(coi1))
plot(randtest(coi2))
```



4. Une illustration


ftp://pbil.univ-lyon1.fr/pub/datasets/CABIOS95/

Please read the file README
it was last modified on Wed Sep 25 17:42:32 1996 - 2813 days ago

Parent directory

Name	Type	Size	Time
.cache	Fichier CACHE	1 KB	28/02/1995 00:00:00
.cache+	Fichier CACHE+	2 KB	22/02/1995 00:00:00
AANames	Fichier	1 KB	12/01/1995 00:00:00
IndiceNames	Fichier	5 KB	05/01/1995 00:00:00
IndiceVals	Fichier	39 KB	05/01/1995 00:00:00
ProtComp	Fichier	54 KB	05/01/1995 00:00:00
ProtNames	Fichier	14 KB	05/01/1995 00:00:00
README	Fichier	4 KB	25/09/1996 00:00:00

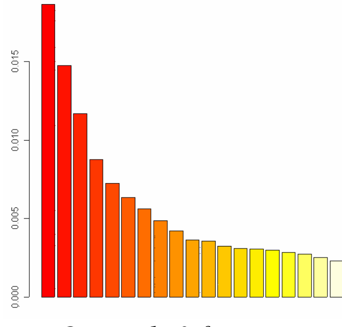
Télécharger les fichiers.

```
read.table("IndiceVals")[1:2,]
     V1   V2   V3   V4   V5   V6   V7   V8   V9  V10  V11  V12  V13  V14  V15
1 4.35 4.38 4.75 4.76 4.65 4.37 4.29 3.97 4.63 3.95 4.17 4.36 4.52 4.66 4.44
2 0.61 0.60 0.06 0.46 1.07 0.00 0.47 0.07 0.61 2.22 1.53 1.15 1.18 2.02 1.95
   V16  V17  V18  V19  V20
1 4.50 4.35 4.70 4.60 3.95
2 0.05 0.05 2.65 1.88 1.32
 x=read.table("IndiceVals")
 dim(x)
[1] 402  20
 x=as.data.frame(t(x))
 scan("IndiceNames",character())[1:5]
Read 402 items
[1] "ANDN920101" "ARGP820101" "ARGP820102" "ARGP820103" "BEGF750101"
 names(x)=scan("IndiceNames",character())
Read 402 items
dim(x)
[1]  20 402

/ftp/ftpdir/pub/datasets/CABIOS95
y=read.table("ProtComp")
y=as.data.frame(t(y))
names(y)=scan("ProtNames",character())
dim(y)
[1]  20 999
row.names(x)=scan("AANames",character())
row.names(y)=scan("AANames",character())
```

y est le tableau d'analyse des correspondances étudié dans Lobry and Gautier (1994) (composition de 999 protéines en acides aminés) :

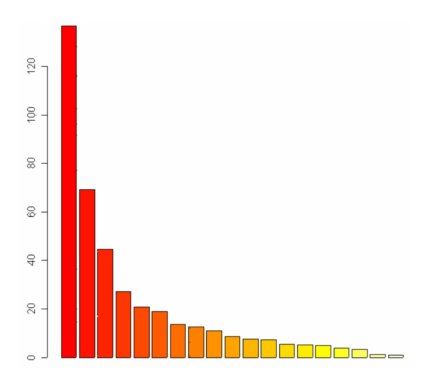
```
coay=dudi.coa(y)
```



On garde 3 facteurs.

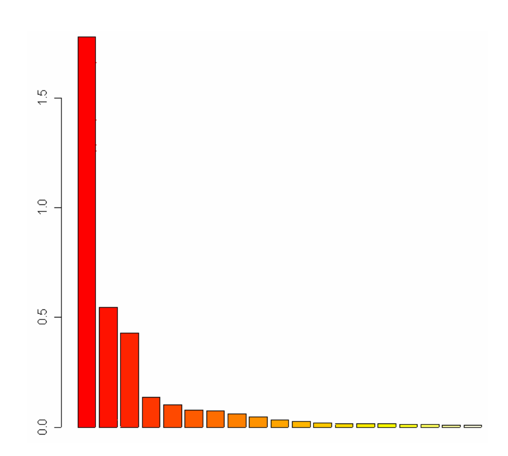
x est un tableau d'ACP normée dans laquelle on introduit la pondération ligne de l'AFC :

```
pcax=dudi.pca(x,row.w=coay$lw)
```



On garde 3 facteurs.

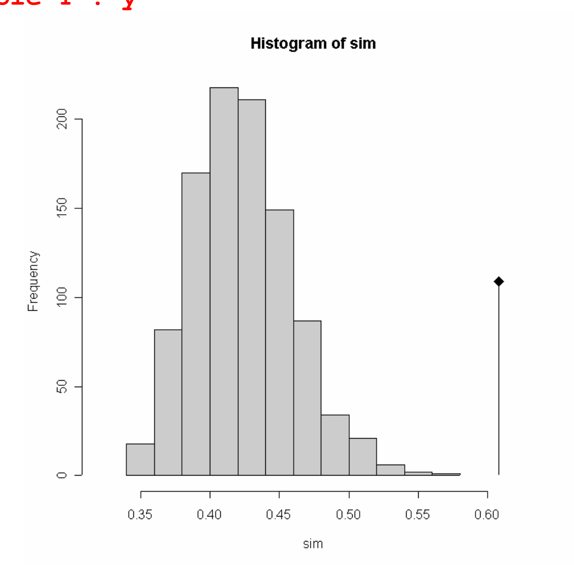
Remarquer la présence de 19 valeurs propres non nulles, le rang d'un tableau d'ACP normée avec $n < p$ valant $n - 1$.




```
coi1=coinertia(pcax,coay)
Select the number of axes: 3
```

On voit un facteur dominant, éventuellement un plan secondaire.

```
plot(randtest(coi1,999,2))
Warning: non uniform weight. The results from permutations
are valid only if the row weights come from the fixed table.
The fixed table is table Y : y
```



Ces résultats sont conformes à l'article de référence.

5. Une remarque

Pour les tableaux très particuliers de la biologie moléculaire, on voudra sans doute éliminer la conservation du tableau croisé. Il suffira de mettre en commentaire quelques lignes de la fonction. Faire une copie locale et la conserver dans un fichier texte :

```
flocal=coinertia
dput(flocal,"flocal.txt")
```

Annuler le calcul du tableau croisé :

```
  # tabcoiner <- t(as.matrix(dudiY$tab)) %*% (as.matrix(dudiX$tab) * dudiX$lw)
  # tabcoiner <- data.frame(tabcoiner)
  # names(tabcoiner) <- names(dudiX$tab)
  # row.names(tabcoiner) <- names(dudiY$tab)
```

Remplacer la ligne qui conserve ce tableau croisé :

```
  # res <- list(tab = tabcoiner, cw = dudiX$cw, lw = dudiY$cw)
  res <- list(tab = NULL, cw = dudiX$cw, lw = dudiY$cw)
```

Sauvegarder et recharger :

```
flocal <- dget("flocal.txt")
 coi3=flocal(pca2,coa2)
Select the number of axes: 2

object.size(coi2)
[1] 184916
object.size(coi3)
[1] 28520
dim(coi2$tab)
[1] 168 111
```

La liste ne contient plus le tableau croisé : certaines fonctions auront des difficultés (**inertia.dudi** par exemple) mais l'essentiel est conservé dans un volume très réduit.
Pour le test de permutation la matrice W de la page 4 permet de conserver un bloc (X ou Y) et de permuter la matrice des produits scalaires de l'autre comme dans le test de Mantel. Le programme en C devra être modifié en conséquence (à faire).

.

Chessel, D., and S. Dolédec. 1993. ADE Version 3.6 : HyperCard © Stacks and Programme library for the Analysis of Environmental Data. Manuel d'utilisation. 8 fascicules URA CNRS 1451, Université Lyon 1, 69622 Villeurbanne cedex.

Culhane, A., G. Perriere, and D. Higgins. 2003. Cross-platform comparison and visualisation of gene expression data using co-inertia analysis. BMC Bioinformatics **4**:59.

Lobry, J., and C. Gautier. 1994. Hydrophobicity, expressivity and aromaticity are the major trends of amino-acid usage in 999 Escherichia coli chromosome-encoded genes. Nucleic Acids Research **22**:3174-3180.

Thioulouse, J., S. Dolédec, D. Chessel, and J. M. Olivier. 1994. ADE Software: multivariate analysis and graphical display of environmental data. Pages 57-62 *in* G. Guariso and A. Rizzoli, editors. Software per l'ambiente. Pàtron Editore, Bologna.

Thioulouse, J., and J. R. Lobry. 1995. Co-inertia analysis of amino-acid physico-chemical properties and protein composition with the ADE package. Computer Applications in the Biosciences **11**:321-329.